**UNIVERSIDAD DE OVIEDO**

|  |
| --- |
| http://www.uniovi.es/documents/31582/23503958/Logo+Universidad+de+Oviedo.jpg/fc0da4a3-441e-4078-9392-d9e700a76b9a?t=1468929363672 |

ESCUELA DE INGENIERÍA INFORMÁTICA

**TRABAJO FIN DE GRADO**

“TITULO DEL TRABAJO FIN DE GRADO”

**TUTOR:**

|  |  |
| --- | --- |
|  | **AUTOR:** |
| **VºBº del Tutor del Proyecto** |  |

Agradecimientos

Esta sección no es en absoluto obligatoria, pero es el lugar correcto para dedicar el proyecto a las personas/instituciones/empresas/… que se desee.

Resumen

Un texto breve (una cara aproximadamente) que describa qué se ha hecho en el proyecto, sus principales objetivos, la utilidad que se le quiere dar, si está destinado a algún cliente real, aspectos sobre la tecnología usada y cosas similares que permitan hacerse una idea rápida del trabajo realizado.

Se trata de describir brevemente todos los aspectos más importantes del proyecto destacando en lo posible sus puntos fuertes para permitir comprenderlo fácilmente en una lectura rápida sin tener más referencias del mismo. Por tanto, no debe ser un texto demasiado largo ni complejo.

Palabras Clave

Palabra1, Palabra2, Palabra3, …

De 5 a 7 palabras[[1]](#footnote-1) clave que mencionen conceptos de capital importancia en el proyecto: Cosas que el proyecto manipula (Ej.: “Máquinas Expendedoras”, “Automóviles”), tecnologías usadas (Ej.: J2EE, RMI), utilidad del proyecto (Ej.: Gestión de Existencias), temática (Ej.: Historia Medieval, Aeronáutica) y cosas similares.

Si finalmente saliesen demasiados términos, conviene hacer una selección de los más relevantes para quedarse con el número indicado.

*Abstract*

Traducción al inglés del resumen anterior. Conviene hacerlo una vez se tenga la versión definitiva de dicho resumen.

*Keywords*

Ídem a la sección anterior.

*Introducción a la norma UNE 157801 y aplicación al PFC.*

La norma 157801 es una norma que tiene por objeto establecer las características que deben ser cubiertas en los proyectos de los Sistemas de Información a realizar, para que satisfagan los fines a los que están destinados. Esta norma tiene su origen en la norma UNE 157001 y puede servir de pauta para proyectos tanto públicos como privados.

Los PFC (Proyectos de Fin de Carrera) son un caso específico de proyecto Informático y como tal, en este documento se presentan los apartados obligados por la norma y de forma adicional, se incorporan también algunos otros elementos particulares y que forman parte de la exigencia de un PFP, como ejemplo más evidente de esto, podemos enumerar la portada oficial con los datos académicos y el resumen junto con las palabras clave.

Según la norma el proyecto consta de los siguientes documentos que en el caso del PFC, por comodidad, pueden incluirse en un volumen único:

* Índice General.
* Memoria.
* Anexos:
  + Documentación de entrada.
  + Análisis y Diseño del Sistema
  + Estimación de Tamaño y Esfuerzos.
  + Planes de Gestión del Proyecto.
  + Plan de Seguridad.
* Especificaciones del Sistema.
* Presupuesto.
* Estudios con Entidad Propia.

Finalmente, la norma es independiente de la metodología, proceso y tecnología usada, así que en este documento se presentan algunos entregables particulares que se consideran importantes para un TFG siguiendo una variante del Proceso Unificado de Desarrollo de Software (RUP).

**Deferencias de esta memoria frente al estándar (Lo que le sobra y lo que le falta)**

Se ha añadido todo lo referente a Implementación y pruebas y manuales de usuario, todo ello a partir del punto 6 de esta memoria. Así mismo también se han añadido algunos apartados más propios del ámbito científico y académico como los aspectos teóricos, las conclusiones o los posibles avances futuros.

Índice General

NOTA: No debemos olvidarnos de generarlo cuando se cierre la documentación, puesto que en caso contrario pueden quedar referencias mal actualizadas o texto erróneo (tanto éste como el de figuras). **BORRAR ESTA NOTA EN LA DOCUMENTACIÓN FINAL**.

[1 Presentación 13](#_Toc192325240)

[2 Aspectos Teóricos 13](#_Toc192325241)

[2.1 Concepto 1 13](#_Toc192325242)

[3 Memoria del Proyecto 14](#_Toc192325243)

[3.1 Hoja de identificación 15](#_Toc192325244)

[3.2 Introducción 16](#_Toc192325245)

[3.3 Objeto 16](#_Toc192325246)

[3.4 Antecedentes 16](#_Toc192325247)

[3.5 Descripción de la situación actual 16](#_Toc192325248)

[3.6 Normas y referencias 17](#_Toc192325249)

[3.6.1 Disposiciones legales y normas aplicadas 17](#_Toc192325250)

[3.6.2 Bibliografía 17](#_Toc192325251)

[3.6.3 Métodos, Herramientas, Modelos, Métricas y Prototipos 18](#_Toc192325252)

[3.6.4 Mecanismos de control de calidad aplicados durante la redacción del proyecto 18](#_Toc192325253)

[3.6.5 Otras referencias 18](#_Toc192325254)

[3.7 Definiciones y abreviaturas 19](#_Toc192325255)

[3.8 Requisitos iniciales 19](#_Toc192325256)

[3.9 Alcance 20](#_Toc192325257)

[3.10 Hipótesis y restricciones 20](#_Toc192325258)

[3.11 Estudios de alternativas y viabilidad 20](#_Toc192325259)

[3.12 Descripción de la solución propuesta 20](#_Toc192325260)

[3.13 Análisis de Riesgos 20](#_Toc192325261)

[3.14 Organización y gestión del proyecto 21](#_Toc192325262)

[3.15 Planificación temporal 21](#_Toc192325263)

[3.16 Resumen del Presupuesto 21](#_Toc192325264)

[4 Anexos 22](#_Toc192325265)

[4.1 Documentación de entrada 22](#_Toc192325266)

[4.2 Análisis y Diseño del sistema 22](#_Toc192325267)

[4.2.1 Análisis 22](#_Toc192325268)

[4.2.2 Diseño 27](#_Toc192325269)

[4.3 Estimación del tamaño y esfuerzo 35](#_Toc192325270)

[4.4 Planes de gestión del proyecto 35](#_Toc192325271)

[4.5 Plan de seguridad 35](#_Toc192325272)

[4.6 Otros documentos que justifiquen y aclaren conceptos expresados en el proyecto 36](#_Toc192325273)

[5 Especificación del Sistema 36](#_Toc192325274)

[5.1.1 Especificación de Casos de Uso 37](#_Toc192325275)

[5.1.2 Análisis de Escenarios 37](#_Toc192325276)

[5.1.3 Actores del Sistema 38](#_Toc192325277)

[6 Presupuesto 39](#_Toc192325278)

[7 Estudios con Entidad Propia 39](#_Toc192325279)

[8 Implementación del Sistema 41](#_Toc192325280)

[8.1 Estándares y Normas Seguidos 41](#_Toc192325281)

[8.2 Lenguaje de Programación 41](#_Toc192325282)

[8.3 Herramientas y Programas Usados para el Desarrollo 41](#_Toc192325283)

[8.3.1 Programa 1 41](#_Toc192325284)

[8.3.2 Programa 2 41](#_Toc192325285)

[8.4 Creación del Sistema 41](#_Toc192325286)

[8.4.1 Problemas Encontrados 41](#_Toc192325287)

[8.4.2 Descripción Detallada de las Clases 42](#_Toc192325288)

[9 Desarrollo de las Pruebas 43](#_Toc192325289)

[9.1 Pruebas Unitarias 43](#_Toc192325290)

[9.2 Pruebas de Integración y del Sistema 43](#_Toc192325291)

[9.3 Pruebas de Usabilidad 44](#_Toc192325292)

[9.4 Pruebas de Rendimiento 44](#_Toc192325293)

[10 Manuales del Sistema 45](#_Toc192325294)

[10.1 Manual de Instalación 45](#_Toc192325295)

[10.2 Manual de Ejecución 45](#_Toc192325296)

[10.3 Manual de Usuario 45](#_Toc192325297)

[10.4 Manual del Programador 45](#_Toc192325298)

[11 Conclusiones y Ampliaciones 47](#_Toc192325299)

[11.1 Conclusiones 47](#_Toc192325300)

[11.2 Ampliaciones 47](#_Toc192325301)

[11.3 Contenido Entregado en el CD-ROM 47](#_Toc192325302)

[11.3.1 Contenidos 47](#_Toc192325303)

[11.3.2 Código Ejecutable e Instalación 50](#_Toc192325304)

[11.3.3 Ficheros de Configuración 50](#_Toc192325305)

[11.4 Índice Alfabético 50](#_Toc192325306)

[11.5 Código Fuente 51](#_Toc192325307)

[11.5.1 Paquete Ejemplo 1: 51](#_Toc192325308)

Índice de Figuras

[Figura 1.1 Ejemplo 15](#_Toc178156747)

[Figura 3.1 Boceto de una interfaz 31](#_Toc178156748)

[Figura 4.1 Ejemplo de arquitectura del sistema 38](#_Toc178156749)

# Presentación

En este apartado se debe incluir una breve presentación con el objetivo de justificar y orientar al lector dando paso a las siguientes secciones donde en primer lugar se introducen los aspectos teóricos a utilizar, para más adelante centrarse en la memoria del desarrollo del proyecto en si mismo.

# Aspectos Teóricos

Aquí se deben describir aspectos tales como conceptos, herramientas y tecnologías existentes que vamos a usar en nuestro proyecto. Conviene limitarse a un máximo de 2 hojas por cada uno de ellos aproximadamente (es una descripción, no un tutorial sobre ello), describiendo por qué y para qué usamos esto en nuestro proyecto, pero sin hacer descripciones muy grandes de características y similares (para ello podemos referenciar enlaces Web, libros, etc. que las describan más detalladamente, que posteriormente podemos incluir en la bibliografía):

* En el caso de conceptos sobre los que trata el proyecto, debemos decir su origen, aplicación e importancia, para tener una idea de lo que se va a tratar. A modo de ejemplo, si usamos bases de datos orientadas a objetos debemos decir que son, que importancia tienen y describir brevemente su forma de trabajo. Si nuestro proyecto es sobre emisión de video en directo, debemos describir la técnica usada para ello y como funciona, etc.
* En el caso de tecnologías debemos decir su creador u origen, para lo que sirve y que aplicaciones ha tenido, para qué la vamos a usar, la versión que hemos empleado, etc.
* En el caso de herramientas debemos decir su fabricante y la utilidad que le vamos a dar dentro de nuestro proyecto.
* Dentro de este apartado también podemos incluir las notaciones empleadas para representar los diagramas del proyecto (como UML) o la metodología usada para el desarrollo de la documentación (como Métrica 3). Este documento está desarrollado usando la metodología Métrica 3, pero de forma adaptada, contemplando solo aquellos apartados que se han considerado más adecuados para un proyecto de fin de carrera. No obstante, conviene hacer una reseña a esta metodología si se usa esta plantilla.

Por último, debemos recordar describir de esta forma todo aquello que usemos en el proyecto y dar especial importancia a todo aquello que sea “novedoso”.

## Concepto 1

Definición

…

# Memoria del Proyecto

El concepto de memoria de un proyecto es, en esencia, un resumen del proyecto para personas que desconozcan o no posean conocimientos avanzados de la naturaleza del proyecto y/o sus tecnologías, o incluso no posean conocimientos específicos de informática. Por tanto, debemos orientarla de manera que cualquier persona pueda entender que se ha hecho durante todo el proyecto.

Los puntos obligatorios de la memoria varían mucho de unos proyectos a otros, pero en este documento se proponen unos mínimos. En muchos casos, la memoria tiene un apartado por cada parte importante del proyecto, por ejemplo (Introducción, Requerimientos, Análisis y Diseño, Presupuesto, etc.) y en cada apartado se resume (para el perfil de lector mencionado anteriormente) el contenido del apartado técnico correspondiente. En cualquier caso, podemos orientar la memoria de la siguiente forma:

## Hoja de identificación

* Título del proyecto y su código identificador, si procede
* Cliente:
  + Razón social, C.I.F., dirección, teléfono y fax de la persona jurídica que ha encargado el proyecto, o si fuera una persona física, nombre y apellidos, N.I.F., dirección, teléfono y fax.
  + Nombre y apellidos, N.I.F., dirección profesional, teléfono y fax del representante legal de la persona jurídica que ha encargado el proyecto.
* Suministrador:
  + Razón social, C.I.F., dirección, teléfono y fax de la persona jurídica que ha recibido el encargo de la realización del proyecto
  + Nombre y apellidos, N.I.F., dirección profesional, teléfono y fax del representante legal de la persona jurídica que ha recibido el encargo de la realización del proyecto.
  + En su caso Nombre y apellidos, titulación, y si procede colegio profesional al que pertenece, número de colegiado, N.I.F., dirección profesional, teléfono, fax y cualquier otro identificador profesional que pueda aparecer o existir, de cada uno de los autores del proyecto.
* Fecha y firma de los anteriormente mencionados
* Resumen
* Duración estimada
* Coste

## Introducción

En este apartado se debe realizar una breve descripción del objetivo, contenido y estructura del documento, a un nivel de detalle que permita que pueda ser comprendido por un lector no experto

## Objeto

En esta sección se debe describir el motivo por el que se desarrolla el proyecto. Se trataría de describir brevemente, en lenguaje no técnico, de que va a consistir el proyecto y el motivo de su desarrollo, incluyendo la descripción de las necesidades que va a cubrir. Se necesita hacer entonces un resumen que describa las aplicaciones del proyecto y que justifique su creación, procurando compararlo favorablemente con otras posibles soluciones.

D:\Archivos de programa\Microsoft Office\MEDIA\CAGCAT10\j0090386.wmf

Figura 1.1 Ejemplo

La imagen de más arriba muestra un ejemplo de cómo se pueden colocar las imágenes en todo el proyecto, incluyendo el estilo de su descripción (que está vinculado con la lista de figuras de más arriba) y la forma de numerar las imágenes propuesta (nº de capítulo – nº de imagen dentro del capítulo).

## Antecedentes

En esta sección deben identificarse y describirse los elementos significativos que han ocurrido en el pasado que tienen influencia en el proyecto.

## Descripción de la situación actual

En esta sección se debe describir el punto de partida del proyecto mostrando los condicionantes de partida sobre los que debe ejecutarse el proyecto. El proyecto debe entenderse como una acción de cambio que permite conducir a la organización desde la situación actual hasta la situación final especificada en el proyecto.

Para lograrlo, se debe hacer una descripción de todos los elementos que condicionen o se vean afectados por el cambio propuesto en el proyecto: recursos humanos con su formación y experiencia, equipamiento hardware, licencias software, etc.

## Normas y referencias

Relación de normas, reglamentos, directrices y otros documentos de referencia que se hayan tenido en cuenta en la elaboración o que se deban tener en cuenta en la ejecución del proyecto. Cuando sea necesario, se deben explicitar en la memoria los puntos más destacables de estas normas. El contenido completo de cada una de las normas puede incluirse como anexo.

### Disposiciones legales y normas aplicadas

En este apartado se debe contemplar el conjunto de disposiciones legales (leyes, reglamentos, ordenanzas, etc.) y las normas que son aplicables al proyecto.

### Bibliografía

En este apartado se debe contemplar el conjunto de libros, revistas, páginas Web u otros textos que el autor considere de interés para justificar las soluciones adoptadas en el proyecto.

#### Libros y Artículos

Libros y artículos usados de alguna forma durante el desarrollo del proyecto o su documentación.

**Formato sugerido:**

**[<PrimerApellidoAutor><DosUltimosDigitosDelAño>]** <Apellidos1, Nombre1; Apellidos2, Nombre2;…>. ”<Título del libro o Articulo>”. <Editorial o lugar de publicación>. <Año (4 cifras)>.

**Ejemplo:**

**[Redondo07]** Redondo L., J. Manuel; De Tal y Cual, Menganito. ”Ejemplo para la plantilla de PFC”. Universidad de Oviedo. 2007.

Si tenemos el ISBN, debemos también ponerlo al final.

#### Referencias en Internet

Páginas Web consultadas para cualquier aspecto relacionado con el desarrollo del sistema o su documentación.

**Formato sugerido:**

**[<PrimerApellidoAutor><DosUltimosDigitosDelAño>]** <Apellidos1, Nombre1; Apellidos2, Nombre2;…>. “<Título de la página Web>”. <URL>. <Año en el que se consultó (4 cifras)>.

**Ejemplo:**

**[Redondo07]** Redondo L., J. Manuel; De Tal y Cual, Menganito. “Título de la página Web de ejemplo”. www.unaurlcualquiera.com. 2007.

Si tenemos más datos que permitan localizar la información dentro de la página, podemos ponerla donde consideremos oportuno.

### Métodos, Herramientas, Modelos, Métricas y Prototipos

En este apartado se debe contemplar la relación de métodos, prototipos, métricas, programas, modelos u otras herramientas utilizadas para desarrollar los diversos cálculos y estimaciones del proyecto.

### Mecanismos de control de calidad aplicados durante la redacción del proyecto

En este apartado se deben enunciar los procesos específicos utilizados para asegurar la calidad durante la redacción del proyecto.

### Otras referencias

En este apartado se deben incluir aquellas referencias que, no estando relacionadas en los apartados anteriores, se consideren de interés para la comprensión y materialización del proyecto.

## Definiciones y abreviaturas

Por orden alfabético, todos los términos que se consideren importantes en la aplicación con una descripción breve de su significado dentro del Sistema.

* **Término1**: Descripción del significado.
* **Término2**: Descripción del significado.

## Requisitos iniciales

Objetivo: Fijar los aspectos externos del producto a construir y utilizarlo como referencia en la validación del producto obtenido especificando el contenido de los entregables tomando como elemento de referencia las características que debe cumplir una vez construido. Los requisitos pueden referirse al producto resultado del proyecto o al proceso a seguir para conseguirlo.

Contenido: El contenido de este capítulo debe depender de las etapas que se incluyan en el proyecto. Se deben desarrollar en las especificaciones del sitema (véase punto 8).

## Alcance

Se trata de describir de nuevo el sistema, pero en lugar de repetir lo que ya hemos dicho de él, tenemos que constatar en este apartado hasta donde vamos a llegar en su construcción, es decir, qué límites vamos a poner en el desarrollo estableciendo qué se va a hacer y qué se va a omitir (en general, hasta donde se va a llegar). Podemos por tanto usar todo lo que hemos dicho en descripciones anteriores para ayudar a describir el alcance del sistema. Conviene dejar claro este apartado para así delimitar la labor de análisis y diseño que vamos a hacer a continuación y evitar así no describir aspectos que se han construido o describir cosas que finalmente no van a construirse.

## Hipótesis y restricciones

En este apartado, se deben identificar y registrar las hipótesis de partida y las restricciones que se han utilizado para la redacción del proyecto y que deberán tenerse en cuenta en la ejecución del mismo. Con este objeto, se realizará una enumeración de las hipótesis de trabajo y restricciones que pueden darse por ciertas y que tengan incidencia en cualquier tipo de estimación reflejada en este documento (coste, plazo, calidad, etc.).

## Estudios de alternativas y viabilidad

En esta sección se deben describir, una por una, todas las alternativas estudiadas. Conviene estudiar 3 o 4 alternativas importantes, salvo que por algún motivo justificado se deba incluir un número menor o mayor de las mismas. En todo caso, siempre es conveniente cuidar de que en esta sección haya un conjunto de sistemas significativo. En función de lo dicho anteriormente, cada sistema podrá dividirse en tres secciones: “Descripción”, “Ventajas” e “Inconvenientes”, aunque es posible cualquier otra división que contenga los aspectos descritos, dependiendo de qué tipo de sistemas se estudien.

#### Sistema 1

#### Sistema 2

## Descripción de la solución propuesta

Describir brevemente la propuesta realizada y las características que permiten valorarla como la más idónea, para ello, la mejor opción consiste en la enumeración de las características significativas de la solución propuesta.

## Análisis de Riesgos

En esta sección se debe identificar los riesgos que afectan al proyecto, tanto en la elaboración del proyecto como en su posterior ejecución de manera que sirva como una propuesta básica del futuro Plan de Riesgos. Para esto, se debe acometer la confección de una lista de riesgos clasificada con una evaluación de sus impactos y posibles acciones de mitigación de riesgos correspondientes.

## Organización y gestión del proyecto

Este apartado tiene el objetivo de explicar cómo se debe llevar a cabo la ejecución del proyecto haciendo partícipes a todas las partes involucradas en el mismo, de la sistemática que se va a utilizar, con el fin de asegurar el éxito del mismo y permitir que cada una de ellas pueda asumir su parte de responsabilidad.

En este apartado deben incluirse todas aquellas normas, directrices o métodos de trabajo que hagan referencia a la organización y gestión del proyecto. Entre ellas cabe destacar sin carácter exclusivo ni completo:

* Organigrama y Matriz de responsabilidades en el proyecto
* Directrices para la gestión de los cambios en el alcance
* Directrices para el seguimiento del proyecto
* Directrices para la recopilación y distribución de información del proyecto
* Directrices de comunicación entre cliente y proveedor
* Directrices a seguir para la aprobación de los entregables
* Lugar donde se realizará el trabajo

## Planificación temporal

En esta sección, se debe describir el plazo de ejecución del proyecto de forma que puedan fijarse las expectativas de quienes van a recibir el producto resultado del mismo.

Debe contener un cronograma explicitando las entregas parciales, hitos intermedios y duración del proyecto a partir de la fecha de iniciación del mismo.

## Resumen del Presupuesto

Explicitar el coste que se ha previsto para la ejecución de forma que pueda tomarse la decisión de proseguir o no con la ejecución de lo valorado. Debe incluir el coste total de la ejecución para la organización que ha de hacerse cargo de este proyecto. En este apartado debe tenerse especial cuidado en presentar las cifras de manera no ambigua, completa, sin costes ocultos y dando un total general desglosado por partidas.

# Anexos

Este apartado está formado por los documentos que tienen como objetivo desarrollar, justificar o aclarar apartados específicos de la memoria u otros documentos básicos del proyecto. Este documento debe contener los anexos necesarios (según proceda en cada caso).

## Documentación de entrada

Este anexo debe incluir la documentación de la que debe disponer el autor del proyecto (Petición de Propuesta, Pliego de Condiciones, información de fases anteriores).

## Análisis y Diseño del sistema

Este anexo debe incluir los documentos de análisis y diseño.

La estructura de este anexo debe depender de la metodología empleada y la complejidad del proyecto, alguno de los aspectos que pueden incluirse en este apartado son:

• En cuanto al análisis: Modelo del sistema a construir, realizado a partir de los requisitos iniciales.

• En cuanto al diseño: Arquitectura del sistema propuesto y los modelos de diseño correspondientes a la funcionalidad, interfaces y datos. Todo ello realizado a partir del modelo de análisis señalado anteriormente.

El desarrollo de este apartado puede ser muy dispar dependiendo del alcance del proyecto, de la metodología empleada, del tamaño del proyecto, etc. En función de éstos, en este apartado se podrán incluir todos aquellos aspectos tales como Análisis de casos de uso, Diseño de alto nivel, Diseño detallado y resultados de las actividades de análisis y diseño con los diagramas que se consideren procedentes.

Como sugerencia, se enumeran a continuación algunos entregables pertenecientes tanto a un posible análisis como a una tentativa de diseño.

### Análisis

Este apartado contendrá toda la especificación de requisitos y toda la documentación del análisis de la aplicación, a partir de la cual se elaborará posteriormente el diseño.

#### Identificación de los Subsistemas en la Fase de Análisis

El objetivo de esta sección es analizar el sistema para poder descomponerlo en sistemas más pequeños (subsistemas) que faciliten su posterior análisis.

#### Descripción de los Subsistemas

En esta sección debemos enumerar todos los subsistemas que identifiquemos inicialmente en la aplicación. Los subsistemas son agrupaciones de paquetes y clases que tienen un objetivo propósito común. Ejemplos de subsistemas pueden ser todas las clases que manejen la base de datos (subsistema “base de datos”), clases que agrupen un conjunto de servicios relacionados, clases del cliente de esos servicios, etc.

#### Descripción de los Interfaces entre Subsistemas

Una vez identificados los subsistemas, debemos también describir cómo será la comunicación entre los mismos. Ejemplos de ello son por ejemplo el uso de un protocolo (IP, SOAP) o una API o Interfaz de operaciones. En general conviene destacar si estos subsistemas se comunicarán localmente (dentro de la propia máquina) o por red.

#### Análisis de Casos de Uso y Escenarios

##### Caso de Uso 1

En esta sección debemos desarrollar un conjunto de diagramas de secuencia que describan la interacción entre los objetos que hemos identificado anteriormente en los casos de uso, escenarios y subsistemas ya vistos.

Estos diagramas de secuencia son de la fase de análisis, por lo que no debemos intentar dar demasiado detalle en ellos, ya que el objetivo es ver las operaciones que habrá entre objetos y el flujo de comunicación general. El objetivo es describir cada operación viendo qué objetos tomarán parte en la misma y como interactuarán entre ellos para lograr el fin de esa operación, para luego darle todo el detalle posible en la fase de diseño. No es necesario dar nombres de métodos reales o especificar todos los parámetros. En esta fase es preferible poner “insertar un usuario” a “insertarUsuario(Usuario)”.

Los diagramas de la fase de diseño se supone que los podemos desarrollar a partir de los que desarrollemos aquí. Se organizarán según los escenarios identificados anteriormente, desarrollando al menos uno por cada uno de ellos. No obstante, si los escenarios representan operaciones muy sencillas o triviales no es necesario hacer un diagrama de secuencia para las mismas. Tampoco tiene mucho sentido desarrollar muchos diagramas de secuencia casi iguales; si varias operaciones funcionan prácticamente de la misma forma, bastaría con indicar que el diagrama correspondiente a la operación X es muy similar al mostrado para la operación Y, indicando en texto las diferencias.

##### Escenario 1.1

No debemos olvidarnos de incluir una explicación del diagrama con aquello que no quede del todo claro en el mismo.

#### Análisis de Clases en la Fase de Análisis

En la fase de análisis podemos identificar ya posibles clases del sistema, a partir de los casos de uso, escenarios y subsistemas ya vistos. Estas clases no tienen porque ser definitivas ni contener todas las operaciones que finalmente contendrán (debemos, eso sí, contemplar las operaciones que aparecen en los diagramas de secuencia para lograr coherencia), pero pueden servir para luego desarrollar más el esquema creado, en la fase de diseño.

Al igual que lo dicho anteriormente para los diagramas de secuencia, estas clases no deben tener nombres de operaciones con sus parámetros exactos, sino más bien una indicación de lo que deben hacer esas operaciones (en lugar de “insertarUsuario(Usuario:u)” es mejor poner simplemente “insertar usuario”).

Las clases deben estar organizadas por paquetes, que también debemos identificar (y justificar). Previamente a describir los paquetes uno por uno, debemos incluir **un diagrama de clases global que muestre la relación entre todas ellas**.

Se recuerda que esta es sólo la fase de análisis y no es necesario lograr un nivel de profundidad de detalle muy elevado, sino que lo que se busca es una idea aproximada (pero precisa) de cómo va a ser el sistema a construir.

##### Paquete 1

Incluimos la descripción del paquete y su utilidad y luego rellenamos una tabla como la siguiente por cada clase del mismo:

|  |
| --- |
| Nombre de la Clase |
|  |
| Descripción | |
|  | |
| Responsabilidades | |
|  | |
| Atributos Propuestos | |
| **NombreAtributo**: Descripción de lo que hace  **NombreAtributo2**: Descripción de lo que hace | |
| Métodos Propuestos | |
| **NombreMétodo**: Descripción de lo que hace  **NombreMetodo2**: Descripción de lo que hace | |

##### Paquete 2

|  |
| --- |
| Nombre de la Clase |
|  |
| Descripción | |
|  | |
| Responsabilidades | |
|  | |
| Atributos Propuestos | |
| **NombreAtributo**: Descripción de lo que hace  **NombreAtributo2**: Descripción de lo que hace | |
| Métodos Propuestos | |
| **NombreMétodo**: Descripción de lo que hace  **NombreMetodo2**: Descripción de lo que hace | |

#### Análisis de Interfaces de Usuario

A la hora de diseñar un interfaz de usuario, debemos cumplir con las normas de comunicación persona-máquina existentes, procurando que el interfaz sea usable, permita manejar el programa de manera eficiente y que no sea propenso a provocar errores en los usuarios.

##### Descripción de la Interfaz

En esta sección debemos crear la especificación de las interfaces entre el usuario y el sistema a construir, incluyendo todos los diferentes tipos de pantallas que van a existir, los cuadros de diálogo o los informes que le proporcionarán al usuario.

En este apartado también es importante identificar posibles grupos de usuarios para así aplicar las pantallas a dichos grupos, así como detallar otros aspectos, como lo que vamos a incluir en las pantallas para cumplir con normas de accesibilidad y usabilidad.

Para los distintos tipos de pantallas, se debe hacer un esquema que muestre la disposición de las mismas, que permita identificar donde irá cada elemento y las diferentes zonas de trabajo. Se muestra un ejemplo con este dibujo:

Dibujo2.wmf

Figura 3.1 Boceto de una interfaz

Otra posible opción para este apartado es diseñar ya las pantallas definitivas sin funcionalidad, solo para ver como quedarán en el producto final (es decir, crear un **prototipo**), lo que tiene la ventaja de poder enseñarle al cliente el aspecto de la aplicación desde un primer momento.

##### Descripción del Comportamiento de la Interfaz

En este apartado debemos especificar cosas como los convenios que vamos a crear para validar la entrada de datos de la aplicación, los mensajes de error que mostraremos y el tipo de ayuda que vamos a proporcionar al usuario.

También incluiremos un diagrama que muestre la navegación que habrá entre las pantallas del programa. Debemos mostrar solo las transiciones entre pantallas y no el contenido de cada pantalla en sí (es decir, que en este diagrama las pantallas pueden ser rectángulos con nombres).

#### Especificación del Plan de Pruebas

En esta sección crearemos y diseñaremos el plan de pruebas de la aplicación y sus funciones, así como todos los mecanismos que utilizaremos para detectar errores y corregirlos ya en la fase de implementación.

Las pruebas contemplarán aspectos tanto de funcionalidad de la aplicación como de aspectos de los usuarios o clientes de la misma.

Se contemplarán tres tipos de pruebas:

* **Pruebas Unitarias**: Una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código, o en este caso una clase individual que cumple con una función concreta. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado. A partir de los casos de uso, los escenarios y clases vistos anteriormente, debemos desarrollar pruebas unitarias que consideremos necesarias y especificar los resultados que se espera encontrar una vez ejecutada la operación sobre cada una de ellas. Es conveniente tabular estas pruebas para su aplicación posterior.
* **Pruebas de Integración**: Las pruebas de integración comprenden verificaciones asociadas a grupos de componentes, verificando que éstos funcionan correctamente cuando estos son ensamblados para cumplir con una función concreta. Para ello, cada escenario debe probarse con el mayor número de entradas posibles (y relevantes) que sea posible, incluyéndose entradas con datos correctos y con datos incorrectos para probar que el sistema reacciona correctamente ante errores de los usuarios. Para elaborar estas pruebas debemos tener en cuenta las características de la aplicación.
* **Pruebas del sistema**: Las pruebas del sistema son pruebas de integración del sistema construido completo, que permiten probar el conjunto de todo el sistema y que sus relaciones con otros sistemas que necesite son correctas, verificando así que todas sus especificaciones funcionales y técnicas se cumplen.

Para elaborar las pruebas debemos desarrollarlas a partir de los casos de uso y escenarios, empleando tablas como las que se muestran a continuación. Estas tablas sirven como ejemplo para pruebas de integración o del sistema. Si resultase más sencillo, puede hacerse con pequeñas tablas independientes para cada caso:

**Caso de Uso 1: Gestionar Usuarios:**

|  |  |
| --- | --- |
| ***Escenario E1.1: Añadir Usuario*** | |
| *Caso de Prueba: CP1.1.1* | |
| **Entrada** | **Resultado Esperado** |
| Añadir un usuario no existente | El sistema posee un usuario más |
| *Caso de Prueba: CP1.1.2* | |
| **Entrada** | **Resultado Esperado** |
| Añadir un usuario que ya existe | El sistema no posee un usuario más y se muestra un dialogo notificándolo |
| *Caso de Prueba: CP1.1.3* | |
| **Entrada** | **Resultado Esperado** |
| Cancelar la Operación | El sistema permanece sin cambios. |

### Diseño

#### Diseño de Casos de Uso Reales

Esta sección se usará para evolucionar y detallar los diagramas de secuencia que hemos creado en el análisis, dividiendo estos diagramas por cada caso de uso y escenario pertinente. La estructura a seguir es la de incluir el diagrama en sí (dibujo) y luego hacer una lista explicando cada uno de los pasos existentes en dicho diagrama que así lo requieran. Esta vez los diagramas si deben incluir nombres de clases, métodos y parámetros reales, con la intención de que puedan trasladarse directamente a la implementación del sistema (el objetivo de un buen diseño). Por este motivo no debemos escatimar en detalles a la hora de desarrollarlos.

Cuando se menciona que los diagramas del diseño son una evolución de los del análisis significa que podemos aprovechar el trabajo ya realizado para su creación, copiando y extendiendo el diagrama correspondiente del análisis para desarrollar el nuevo.

En cuanto al número de diagramas a contemplar, al igual que se dijo anteriormente no es necesario incluir todos y cada uno de los posibles, sino que podremos ahorrar describir algunos casos de uso en los siguientes casos:

* Las operaciones sean muy simples (en cuyo caso se puede optar por no decir nada al respecto o colocar simplemente un texto explicativo).
* La operación sea muy similar o idéntica a una que ya está desarrollada en esta sección. En este caso lo mejor es poner en el diagrama original a que otros casos se aplica o representa.

##### Caso de Uso 1

###### Escenario 1.1

**Diagramas de Estados de las Clases**

En caso necesario, si es conveniente dar más detalles de cómo se comporta un determinado escenario, podemos incluir diagramas de estados de las clases involucradas en el mismo para indicar las distintas fases por las que una clase pasará durante el mismo.

##### Caso de Uso 2

###### Escenario 2.1

#### Diseño de Clases

En esta sección representaremos diagramas que muestren los paquetes (y la relación existente entre ellos) y las clases que formarán parte de la implementación final del sistema (incluyendo también las relaciones existentes entre ellas), explicando todo aquel dato acerca de la utilidad de los mismos y justificando todo aquello que consideremos necesario.

* Debemos extraer la información de métodos y atributos a incluir en cada clase (así como las relaciones existentes entre ellas) de los diagramas que hemos desarrollado en el punto anterior.
* También debemos mostrar las clases adicionales que pueden ser necesarias en función de lo que hemos desarrollado anteriormente.
* Debemos mostrar todas las asociaciones y agregaciones (en general, relaciones) que necesitemos, en función de dichos diagramas.
* El conjunto de atributos de las clases debemos crearlo según los métodos, relaciones y en general de las necesidades de cada clase del sistema.
* La jerarquía de clases debemos pensarla de acuerdo a las necesidades de cada subsistema y aplicación.
* Debemos aclarar todos aquellos aspectos que no queden claros en el diagrama, usando notas.
* En general, debemos tener en cuenta que el diseño es una evolución del análisis, por lo que las clases que incluyamos en el diseño deben corresponderse razonablemente con todo el trabajo hecho en esa parte del documento.

##### Diagrama de Clases

Es necesario mostrar por lo menos un diagrama global de clases. Si el diagrama de clases global fuese muy grande, podemos dividir el diagrama en varios diagramas más pequeños según la estructura de la aplicación, mostrando el principal abreviado. Otra opción es hacer que la hoja que tenga el diagrama se imprima en otro formato que nos proporcione más espacio (en horizontal, A3, …).

##### Jerarquía de Paquetes

Diagrama que muestre la jerarquía de paquetes de la aplicación.

#### Diseño de la Base de Datos

Esta sección es la indicada para describir todo lo relativo al sistema de gestión de bases de datos que vamos a usar en nuestra aplicación (si usamos alguno). Esto incluye hablar del sistema en sí, la versión utilizada, las clases empleadas al usarlo, como se integra en el sistema y el diagrama E-R de las tablas y relaciones creadas, todo ello en los apartados que se muestran a continuación.

##### Descripción del SGBD Usado

##### Integración del SGBD en Nuestro Sistema

##### Diagrama E-R

#### Arquitectura del Sistema

El sistema creado puede estar compuesto por varios procesos que colaboran para llevar a cabo la tarea encomendada y esta sección es la indicada para representar estos procesos y la relación existente entre ellos. Debemos ofrecer un diagrama (un ejemplo de cómo podemos hacerlo se muestra a continuación) y posteriormente una descripción de que es cada parte de la aplicación y su función (relacionándolo con las clases y paquetes ya desarrollados).

Dibujo1.wmf

Figura 4.1 Ejemplo de arquitectura del sistema

##### Subsistema 1

##### Subsistema 2

##### Diagramas de Actividades

Si hay alguna operación o funcionalidad dentro del sistema que merezca la pena destacar se documentará mediante un diagrama de actividades. Puede haber tantos como consideréis necesarios.

#### Diseño de la Interfaz

Esta sección debe mostrar ya la interfaz definitiva de la aplicación (que evidentemente deberá ser una evolución del diseño mostrado en el análisis) y las diferentes partes de las que consta. Como es más que probable que todas las pantallas tengan elementos de interfaz comunes, esta es la sección donde se va a hablar de cada uno de esos elementos, su propósito y su función (barra de menús, barras de estado, etc.).

En caso de no haber hecho un prototipo en la fase de análisis (es decir, mostrado el aspecto real del programa), debemos hacerlo aquí. En todo caso, cualquier cosa que quedase pendiente en la fase de análisis acerca de la descripción del interfaz debe concretarse aquí.

#### Especificación Técnica del Plan de Pruebas

El proceso de pruebas se extiende durante todo el proceso de construcción del software y por tanto en esta sección debemos describir cómo vamos a aplicar las pruebas diseñadas y especificar más otro tipo de pruebas que vamos a realizar al software.

No debemos olvidarnos mencionar entonces bajo que máquina (incluyendo sistema operativo y entorno de desarrollo) se realizan las pruebas, y en qué orden se realizan las diferentes pruebas contempladas, entre otros aspectos.

**NOTA**: Es perfectamente posible que, aunque un sistema conste de múltiples procesos independientes que colaboran entre sí (por ejemplo un cliente y un servidor), las pruebas se hagan en sólo una máquina (Ej.: cliente y servidor que se conectan a *localhost*).

##### Pruebas Unitarias

Esta sección contendrá la descripción de la aplicación de las pruebas unitarias que hemos descrito anteriormente (qué datos se van a introducir finalmente sobre qué clases, ahora que conocemos finalmente todas las clases que se van a implementar gracias al diseño), indicando cuando las vamos a realizar y que medidas tomaremos en caso de encontrar fallos. También debemos hablar de si hemos dividido las pruebas entre los diferentes subsistemas de alguna forma.

En este apartado debemos también describir las posibles **pruebas de regresión** que queramos implementar. Estas pruebas de regresión se deben automatizar para poder pasarlas periódicamente cada vez que hagamos cambios importantes en el sistema.

**Se recomienda encarecidamente el empleo de una herramienta de automatización de estas pruebas** unitarias para facilitar mucho esta labor. Posibles herramientas son *JUnit* (Java) o *NUnit*. Realmente esta sería la forma más adecuada de afrontar esta tarea.

##### Pruebas de Integración y del Sistema

Las pruebas funcionales y del sistema que hemos especificado en el análisis deben aplicarse para garantizar que el sistema funciona correctamente. Debemos describir así cómo y cuándo vamos a aplicar este tipo de casos de prueba dentro del sistema, así como las entradas y salidas de estas pruebas. No debemos olvidar que la aplicación de las pruebas y qué ocurre cuando se pasan irá en una sección posterior.

Debemos recordar pues que, tanto en esta sección como en la anterior, ya tenemos las pruebas diseñadas en la fase de análisis, y que ahora se trataría de describir cómo y cuando las vamos a aplicar, qué datos vamos a introducir en cada caso y qué vamos a hacer en caso de acierto o fallo de las pruebas. No se trata de repetir lo mismo de nuevo.

A modo de guía, debemos pensar que lo que diseñemos aquí debe ser directamente aplicable sobre el código de la aplicación.

##### Pruebas de Usabilidad

Dado que es posible que el lector de este documento no esté familiarizado con este tipo de pruebas, haremos una descripción un poco más en detalle de en qué consisten. Este tipo de pruebas determinan la satisfacción del cliente con el producto final y por tanto debemos darle mucha importancia a las mismas, no dejándolas de lado en ningún caso.

Las pruebas de usabilidad tratan de evaluar la aplicación en su funcionamiento real. Debemos realizar un conjunto de pruebas para verificar que las distintas partes del programa se pueden usar adecuadamente. Los objetivos son:

* Mejorar la calidad de la interacción de los usuarios con nuestra aplicación.
* Reducir el tiempo necesario para hacer las distintas tareas en la aplicación y de esta forma aumentar la productividad.

Debemos tener los siguientes elementos en cuenta, para describirlos completamente en esta sección antes de hacer las pruebas en sí:

* **Usuarios**: Distintos grupos de usuarios de la aplicación sobre los que vamos a realizar las pruebas. Cada usuario tendrá un perfil de uso distinto y tenemos que tenerlo en cuenta a la hora de hacer las pruebas.
* **Lugar de realización**: En un laboratorio o bien en la propia casa u oficina del cliente.
* **Metodología**: Describir que vamos a hacer en estas pruebas (el conjunto de pasos a seguir en las mismas).

###### Diseño de Cuestionarios

###### Cuestionario de Evaluación

Debemos elaborar un cuestionario para que los usuarios lo hagan y determinar así distintos aspectos del mismo y de su interacción con la aplicación. Los puntos a tocar son (esto es un esquema que puede ampliarse si se desea para adaptarlo a la aplicación en sí):

* **1º**: **Preguntas de carácter general** a través de las cuales se determine el tipo de usuario y su nivel de conocimiento informático.
* **2º**: **Actividades guiadas** para hacer con nuestra aplicación.
* **3º**: **Batería de preguntas cortas** con los distintos aspectos de la aplicación que se pretendan evaluar.
* **4º**: **Observaciones**, para que el usuario aporte todo lo que considere oportuno de nuestra aplicación.

Cuestionario para el Responsable de las Pruebas

Además, debemos desarrollar un cuestionario para que el responsable de las pruebas pueda anotar distintos aspectos que observe durante la realización de las pruebas.

Un posible desarrollo de todos estos aspectos se describe a continuación.

###### Actividades de las Pruebas de Usabilidad

Preguntas de carácter general

Se muestra un esbozo de un posible cuestionario, que debemos desarrollar y adaptar a nuestras necesidades:

|  |
| --- |
| ¿Usa un ordenador frecuentemente? |
| 1. Todos los días 2. Varias veces a la semana 3. Ocasionalmente 4. Nunca o casi nunca |
| ¿Qué tipo de actividades realiza con el ordenador? |
| 1. Es parte de mi trabajo o profesión 2. Lo uso básicamente para ocio 3. Solo empleo aplicaciones estilo Office 4. Únicamente leo el correo y navego ocasionalmente |
| ¿Ha usado alguna vez software como el de esta prueba? |
| 1. Sí, he empleado software similar 2. No, aunque si empleo otros programas que me ayudan a realizar tareas similares 3. No, nunca |
| ¿Qué busca Vd. Principalmente en un programa? |
| 1. Que sea fácil de usar 2. Que sea intuitivo 3. Que sea rápido 4. Que tenga todas las funciones necesarias |

Actividades guiadas

Hacer un compendio de actividades que se puedan hacer con la aplicación para que nuestros usuarios las hagan y comenten los problemas y dificultades que según su opinión presenta la aplicación (si existiesen). Debemos recoger estas opiniones en el documento. Posibles actividades a probar son:

* Añadir un usuario a la aplicación
* Eliminar un artículo de la aplicación
* …

Preguntas Cortas sobre la Aplicación y Observaciones

Un posible cuestionario de preguntas cortas (a desarrollar más en cada proyecto) es el siguiente:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Facilidad de Uso** | **Siempre** | **Frecuentemente** | | **Ocasionalmente** | | **Nunca** |
| *¿Sabe donde está dentro de la aplicación?* |  |  | |  | |  |
| *¿Existe ayuda para las funciones en caso de que tenga dudas?* |  |  | |  | |  |
| *¿Le resulta sencillo el uso de la aplicación?* |  |  | |  | |  |
| **Funcionalidad** | **Siempre** | **Frecuentemente** | | **Ocasionalmente** | | **Nunca** |
| *¿Funciona cada tarea como Vd. Espera?* |  |  | |  | |  |
| *¿El tiempo de respuesta de la aplicación es muy grande?* |  |  | |  | |  |
| **Calidad del Interfaz** | | | | | | |
| **Aspectos gráficos** | **Muy Adecuado** | | **Adecuado** | **Poco Adecuado** | **Nada Adecuado** | |
| *El tipo y tamaño de letra es* |  | |  |  |  | |
| *Los iconos e imágenes usados son* |  | |  |  |  | |
| *Los colores empleados son* |  | |  |  |  | |
| **Diseño de la Interfaz** | | | **Si** | **No** | **A veces** | |
| *¿Le resulta fácil de usar?* | | |  |  |  | |
| *¿El diseño de las pantallas es claro y atractivo?* | | |  |  |  | |
| *¿Cree que el programa está bien estructurado?* | | |  |  |  | |
| **Observaciones** | | | | | | |
| Cualquier comentario del usuario | | | | | | |

Evidentemente debemos extender este cuestionario de ejemplo para adaptarlo a nuestras necesidades.

Cuestionario para el Responsable de las Pruebas

|  |  |
| --- | --- |
| **Aspecto Observado** | **Notas** |
| *El usuario comienza a trabajar de forma rápida por las tareas* |  |
| *Tiempo en realizar cada tarea* |  |
| *Errores leves cometidos* |  |
| *Errores graves cometidos* |  |
| *<Incluir otras cosas>* |  |
| *…* |  |
| *…* |  |

Los resultados de las pruebas se deben analizar conjuntamente para así establecer una conclusión respecto a cuatro aspectos de usabilidad:

1. **Facilidad de aprendizaje**: Capacidad para aprender la funcionalidad de la aplicación desarrollada y desarrollar las tareas de manera adecuada, midiendo cuanto se tarda en hacer las distintas tareas.
2. **Eficiencia**: Cuanto mejora la labor de los usuarios por usar la aplicación respecto a lo que se hacía anteriormente.
3. **Errores**: Cuantos errores cometen los usuarios en las distintas tareas, lo que decrementa la usabilidad del mismo.
4. **Satisfacción del usuario**: Impresión general de los usuarios al usar la aplicación.

##### Pruebas de Rendimiento

El sistema desarrollado consumirá una determinada cantidad de recursos (memoria y tiempo de proceso) que debemos procurar que sean los menores posible. Por ello, la aplicación debe al menos medirse para ver cuántos recursos consume y se debe intentar eliminar posibles cuellos de botella en el rendimiento que se puedan presentar en uno o varios subsistemas de la misma. Debemos diseñar medios para hacer estas mediciones en nuestra aplicación.

Para esta tarea debemos pues medir el tiempo que tardan las operaciones más importantes que haga el sistema y comprobar si es posible mejorarlas de algún modo una vez que el sistema esté en funcionamiento. En este apartado debemos diseñar que actuaciones vamos a llevar a cabo para hacer estas pruebas y las actuaciones que vamos a llevar a cabo para mejorar el rendimiento en aquellos puntos en los que encontremos problemas.

Una consideración a tener en cuenta es determinar qué actividades serán las más frecuentes y si su rendimiento es adecuado o no, ya que por esta vía conseguiremos una mejor optimización de la aplicación.

En el caso de que el programa deba tener algún requisito respecto al tiempo que debe tardar o a la memoria ocupada, debe tenerse especial cuidado en este aspecto.

Por último, también debemos diseñar pruebas de carga para determinar cómo responde el sistema con un alto número de usuarios o un gran volumen de datos.

## Estimación del tamaño y esfuerzo

Es este anexo debe detallar y estimar cuantas métricas sean de aplicación y de interés al proyecto en decisión de sus autores. El contenido de este anexo debe servir de base para la elaboración del presupuesto detallado.

Se debe determinar por los autores la o las métricas a aplicar al proyecto y se debe valorar cada una de ellas de acuerdo a los datos contenidos en el proyecto y usando los criterios estándar determinados por las instituciones de normalización de métricas.

Si se utilizasen métricas propias deben estar adecuadamente documentadas, contrastadas y referenciadas.

## Planes de gestión del proyecto

Este anexo, según el alcance del proyecto y la complejidad, debe describir la forma en la que se realizará la gestión del proyecto.

Acorde con las normas internacionales, los planes de este tipo deben cubrir aspectos tanto de organización como de gestión del proyecto. Aunque el detalle de estos planes debe estar en concordancia con la complejidad del proyecto, se pueden considerar los siguientes:

• Gestión de la integración

• Gestión del alcance.

• Gestión de plazos.

• Gestión de costes.

• Gestión de la calidad.

• Gestión de recursos humanos.

• Gestión de comunicaciones.

• Gestión de riesgos.

• Gestión de adquisiciones.

## Plan de seguridad

En este anexo se debe definir la seguridad de las entidades y elementos relacionados con el proyecto y las medidas que se deben utilizar para integrar ésta dentro del proyecto.

Sin ser excluyente, este apartado debe contener el Plan de Seguridad, metodologías y herramientas a utilizar en la gestión de la seguridad en el proyecto y la identificación de los puntos críticos donde la seguridad es determinante o está impuesta por ley.

En este apartado se definirían todos los aspectos técnicos, organizativos y legales de la gestión de la seguridad.

## Otros documentos que justifiquen y aclaren conceptos expresados en el proyecto

Se pueden incluir entre otros:

• Catálogos de los elementos constitutivos del objeto del proyecto.

• Listados.

• Información en soportes lógicos, magnéticos, ópticos u otros.

# Especificación del Sistema

Esta sección debe incluir la especificación detallada de los requisitos, incluidos los diagramas que fueran necesarios. La estructura de este documento dependerá de la metodología empleada y la complejidad y alcance del proyecto. Debe cubrir la especificación de requisitos funcionales y no funcionales del producto. Este documento debe servir de base para el análisis y diseño del sistema.

En concreto se propone un documento detallado que se crea para su aprobación formal, es decir, que los potenciales clientes deben ver a partir de él las especificaciones completas del sistema. Además, esta sección construirá una base para solicitar cambios en los requisitos antes de avanzar más en la construcción del sistema.

Los requisitos del sistema se deben mostrar en una tabla como la que se presentará a continuación con ejemplos, ordenados por algún criterio lógico en función de a lo que se refieran. Por ejemplo, si tenemos usuarios tiene sentido agrupar todos los requisitos que tengan que ver con los usuarios.

Tampoco es necesario crear una única tabla para todos los requisitos, pueden crearse varias tablas que agrupen los requisitos que se refieran a una entidad.

Este apartado debe incluir también antes de la tabla de requisitos, si existe como tal, la **especificación textual** que el cliente nos proporcione sobre la aplicación, fruto de las reuniones que hayamos tenido con él o de las entrevistas que podamos haber llevado a cabo. Ha de tenerse en cuenta que esta información es la que usaremos para extraer los requisitos de la aplicación, por lo que no debe faltar sin contamos con ella.

|  |  |  |
| --- | --- | --- |
| **Código Requisito** | **Nombre Requisito** | **Descripción del Requisito** |
| R1.1 | Insertar Usuario | Se debe añadir un usuario al sistema una vez leídos y validados sus datos. |
| R1.2 | Leer Datos Usuario | Deben pedirse los datos completos de un usuario del sistema |
|  |  |  |

Esta tabla puede usarse para todos los **requisitos funcionales** (lo que la aplicación debe hacer), pero debemos especificar también otro tipo de **requisitos no funcionales**, de los que a continuación se muestran ejemplos típicos:

* **Requisitos de Usuario**: Si exigimos al usuario algún tipo de conocimiento previo para manejar la aplicación o alguna de sus partes, debemos especificarlo aquí.
* **Requisitos Tecnológicos**: Si el programa establece que debe funcionar con una versión concreta de un determinado programa o sistema, o bien en un entorno o sistema operativo concreto debemos también hacerlo constar.
* **Requisitos de Usabilidad**: Normas de usabilidad que la aplicación debe cumplir obligatoriamente (nosotros podemos especificar requisitos adicionales a los que el cliente solicite y desarrollarlos en la sección correspondiente).
* **Requisitos de Seguridad**: Si debemos implementar algún tipo de medida de seguridad en el sistema (encriptación de datos, etc.).
* **Requisitos de Tiempo de Respuesta**: Si el sistema debe proporcionar una respuesta en un tiempo acotado.

### Especificación de Casos de Uso

A partir de los requisitos, se deben elaborar los distintos casos de uso de la aplicación teniendo en cuenta las operaciones que debemos hacer y las entidades (actores, otros sistemas) involucradas.

Esta sección debe incluir el clásico diagrama de casos de uso junto con una breve explicación de qué es cada caso de uso y lo que incluye. Si el número de casos de uso fuese muy elevado, se pueden crear múltiples diagramas para que quede todo de forma más clara.

### Análisis de Escenarios

A partir de los casos de uso se podrá desarrollar cada uno de ellos en múltiples escenarios que los detallen mucho más, incluyendo de esta manera todas las posibles operaciones involucradas en cada caso de uso. Para desarrollar los distintos escenarios podemos organizarlos según el caso de uso al que pertenezcan, usando esta tabla para ello:

#### Caso de Uso 1

|  |  |
| --- | --- |
| **Escenario 1.1** | |
| *Precondiciones* |  |
| *Poscondiciones* |  |
| *Excepciones* |  |
| *Iniciado por* |  |
| *Finalizado por* |  |
| *Descripción* |  |
| *Corresponde al Requisito:* |  |

#### Caso de Uso 2

|  |  |
| --- | --- |
| **Escenario 2.1** | |
| *Precondiciones* |  |
| *Poscondiciones* |  |
| *Excepciones* |  |
| *Iniciado por* |  |
| *Finalizado por* |  |
| *Descripción* |  |
| *Corresponde al Requisito:* |  |

Al final de esta sección, podemos incluir una tabla que haga corresponder (vincular) los casos de uso que hemos identificado con el escenario al que pertenecen, como ésta:

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso 1** | **Caso de Uso 2** |
| *Escenario 1.1* | **X** |  |
| *Escenario 1.2* | **X** |  |

Al final de esta sección, también podemos incluir una tabla que haga corresponder (vincular) los casos de uso que hemos identificado (y por tanto los escenarios de cada uno de ellos) a los requisitos dados anteriormente, similar a ésta:

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso 1** | **Caso de Uso 2** |
| *Requisito 1* | **X** |  |
| *Requisito 2* |  | **X** |

El objetivo de esta tabla es mostrar de formar directa y visual el motivo por el que se ha desarrollado cada caso de uso (que requisito justifica su inclusión).

### Actores del Sistema

Una vez que se hayan identificado en los casos de uso, debemos describir los actores que aparecen durante el proceso de análisis: qué hacen, con qué o quién interactúan y el motivo por el que están incluidos en el sistema.

A la hora de identificar actores debe analizarse la aplicación y sus usos. Ejemplos de posibles actores son: Usuario anónimo, usuario registrado, administrador,...

# Presupuesto

En esta sección debemos rellenar esta tabla Excel para calcular el coste económico de desarrollar el sistema planteado en la documentación. Debemos considerar cosas como las horas de análisis, diseño y programación, el personal necesario, materiales, programas, equipos, etc.



Si fuese necesario, podemos incluir en esta sección una planificación temporal de las distintas etapas del proyecto (análisis, diseño, etc.).

El Presupuesto debe contener:

• Cuando proceda, un cuadro de precios de las unidades de medida correspondientes: componentes de hardware, elementos de software, horas persona de diferentes categorías, elementos auxiliares y otros.

• Cuando proceda, costes de unidades lógicas con entidad propia dentro del proyecto, con la descomposición correspondiente de componentes de hardware, elementos de software, horas persona, elementos auxiliares y otros.

• El presupuesto propiamente dicho debe contener la valoración económica global, descompuesta siguiendo la estructura de desglose de los elementos utilizada en la planificación y ejecución del proyecto.

• El presupuesto debe especificar claramente las bases con las que se confecciona el mismo.

# Estudios con Entidad Propia

Este apartado tiene como misión incluir los documentos convenientes para adecuar el proyecto a las exigencias legales y que no aparezcan en los apartados anteriores.

Puede comprender, entre otros y sin carácter limitativo estudios relativos a:

• Legislación sobre Seguridad y Protección de Datos.

• Legislación sobre Propiedad Intelectual e Industrial

• Prevención de Riesgos Laborales.

• Impacto Ambiental.

# Implementación del Sistema

## Estándares y Normas Seguidos

Descripción breve de los estándares y normas que hayamos usado en nuestra aplicación a la hora de desarrollar su código y si nos hemos ocupado de validar que esos estándares se cumplan efectivamente.

## Lenguaje de Programación

Descripción del lenguaje de programación usado, la versión y distribución, los módulos o complementos del mismo empleados, etc.

## Herramientas y Programas Usados para el Desarrollo

Descripción de todas las herramientas de desarrollo (Eclipse, Visual Studio, etc.), sistemas adicionales existentes, complementos y otros productos software que necesitemos para la implementación de nuestro sistema. Debemos dejar claro que versión usamos, para qué y cómo interactuará con nuestro sistema.

### Programa 1

### Programa 2

## Creación del Sistema

Todos los aspectos con los que nos hemos encontrado durante la implementación debemos describirlos aquí.

### Problemas Encontrados

Enumeramos los problemas encontrados en el desarrollo y la solución que le hemos dado, si hubo alguno que merezca la pena destacar.

### Descripción Detallada de las Clases

En esta sección debemos describir las clases más relevantes de nuestra aplicación de manera que se detallen todos sus atributos y métodos así como su descripción y función exacta, tal y como aparecen en el código de la aplicación. Para hacerlo lo mejor posible debemos elaborar esta descripción una vez cerrado el desarrollo de la aplicación en sí, para evitar problemas de versiones por revisiones al código que surjan durante el desarrollo. Para ello podemos hacer dos cosas:

* Emplear una herramienta que vuelque las clases y sus comentarios a un formato que podamos usar en este documento y que contemple todos los aspectos vistos antes (es decir, que el resultado sea similar a la tabla mostrada más abajo). Herramientas como Javadoc o similares son adecuadas para este fin.
* En caso de no disponer o no querer usar una herramienta como la mencionada anteriormente, empleamos la tabla que se muestra a continuación.

Para cada clase se pondrá un título de nivel 4 y se comenzará en una página nueva. NO rellenar con saltos de párrafo (return).

#### Clase 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Tipo** | **Descripción** | **Hereda de…** |
|  | Abstracta, Final,… | ¿Para qué sirve? |  |
| ***Responsabilidades*** | | | |
| **Número** | **Descripción** | | |
|  |  | | |
|  |  | | |
| ***Métodos*** | | | |
| **Acceso|Modo** | **Tipo de Retorno** | **Nombre** | **Parámetros y tipos** |
| (Público, Protegido, Privado)|(Abstracto, Virtual, Estático) |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| ***Atributos*** | | | |
| **Acceso** | **Modo** | **Tipo o Clase** | **Nombre** |
| (Público, Protegido, Privado) | (Estático, Final) |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **Observaciones** | | | |
| Cualquier comentario adicional a la clase... | | | |

En caso de ser un proyecto hecho en parejas, conviene indicar adicionalmente en estas tablas quien ha sido el autor de esta clase.

# Desarrollo de las Pruebas

## Pruebas Unitarias

Describir aquí el resultado de las pruebas unitarias que ya hemos diseñado y descrito anteriormente, así como los resultados obtenidos y las actuaciones que hemos llevado a cabo ante los errores y problemas encontrados.

## Pruebas de Integración y del Sistema

Ejecutamos las pruebas funcionales ya diseñadas y anotamos el resultado que obtenemos, comparándolo con el que especificamos anteriormente. Podemos hacerlo a partir de una tabla modificada de la anterior como ésta (si es más sencillo, puede hacerse con pequeñas tablas independientes para cada caso):

|  |  |
| --- | --- |
| ***Escenario E1.1: Añadir Usuario*** | |
| *Caso de Prueba: CP1.1.1* | |
| **Entrada** | **Resultado Esperado** |
| Añadir un usuario no existente | El sistema posee un usuario más |
|  | **Resultado Obtenido** |
|  | El sistema efectivamente posee un usuario más |
| *Caso de Prueba: CP1.1.2* | |
| **Entrada** | **Resultado Esperado** |
| Añadir un usuario que ya existe | El sistema no posee un usuario más y se muestra un dialogo notificándolo |
|  | **Resultado Obtenido** |
|  |  |
| *Caso de Prueba: CP1.1.3* | |
| **Entrada** | **Resultado Esperado** |
| Cancelar la Operación | El sistema permanece sin cambios. |
|  | **Resultado Obtenido** |
|  |  |

## Pruebas de Usabilidad

A partir de los cuestionarios que se diseñaron anteriormente y de los procedimientos explicados, mostramos aquí el resultado de aplicarlos sobre todos los usuarios que hayamos empleado para estas pruebas.

## Pruebas de Rendimiento

Explicamos el desarrollo, resultados y cambios derivados de la ejecución de todas las pruebas de rendimiento que hayamos especificado para el sistema, según lo que hemos diseñado en el anteriormente.

# Manuales del Sistema

## Manual de Instalación

Elaborar un manual que contemple todos los pasos necesarios para instalar nuestro sistema, incluyendo la instalación de otras herramientas o software cualquiera (sea o no comercial) necesario para que funcione. Debemos explicarlo todo paso a paso de forma clara y acompañarlo por capturas de pantalla adecuadas.

## Manual de Ejecución

Este manual contemplará todos los pasos necesarios para el arranque de nuestro sistema, lo que es especialmente importante en caso de sistemas con clientes y servidores o distintos procesos que deban arrancarse independientemente.

Por otra parte, también debemos incluir procedimientos para parar adecuadamente la aplicación.

## Manual de Usuario

El manual de usuario es algo muy importante debido a que es el documento que servirá a los usuarios de nuestro sistema para saber cómo funciona cada una de las partes de nuestra aplicación. Debemos pues describir cómo funcionan todas las opciones de las misma, que parámetros tiene, que cosas debemos hacer para que todas las operaciones funcionen correctamente y cualquier otro aspecto que consideremos oportuno para explicar el funcionamiento del sistema.

No debemos escatimar detalles en este manual ya que es la herramienta para que los usuarios comprendan nuestro sistema. También debemos hacer el mayor uso posible de capturas de pantalla para mejorar nuestras explicaciones.

## Manual del Programador

En este manual debemos describir cualquier aspecto que pueda ayudar a otros programadores a ampliar, modificar o entender aspectos de la construcción de nuestra aplicación. Debemos por tanto hacer una descripción general de los distintos aspectos involucrados en la construcción del sistema que puedan ser más difíciles de entender y también describir los procedimientos necesarios para hacer ciertas ampliaciones que hayamos contemplado en el diseño del sistema (añadir nuevas entidades, nuevos atributos a entidades existentes, nuevos servicios que usen a los ya desarrollados, modificaciones en la interfaz, etc.).

# Conclusiones y Ampliaciones

## Conclusiones

Conclusiones del sistema: Qué hemos elaborado, si los resultados están dentro de lo esperado, si hemos cumplido las expectativas, justificación de haber escogido las mejores opciones para cada uno de los aspectos del sistema, etc.

## Ampliaciones

Cualquier labor de ampliación que tengamos contemplada en el sistema debe ser descrita aquí, mencionando en qué consiste, cómo ampliará el sistema, qué ventajas nos aporta y porqué no se ha incluido en el sistema diseñado, entre otros aspectos.

## Contenido Entregado en el CD-ROM

### Contenidos

Descripción del contenido de los diskettes o CDs (directorios y para qué sirve cada cosa), descripción de esta documentación y de cualquier material que adicionalmente se entregue en la presentación. En esta sección se presenta una estructura de directorios de ejemplo para el CD que se puede seguir para distribuir todos sus contenidos en el mismo. Conviene por tanto tenerla en cuenta desde el principio de la implementación.

##### Introducción

La estructura de directorios del proyecto debe poder recoger todos los ficheros relacionados con el proyecto, clasificándolos por su propósito dentro del mismo. Los tipos más frecuentes son: ficheros fuente, ficheros de configuración, ficheros de documentación…

Se deben crear directorios para contener cada uno de los tipos de ficheros. Tener una estructura estandarizada de los directorios del proyecto es importante por varias razones:

* Ayuda a localizar la información del proyecto. Por ejemplo, los ficheros fuente siempre deben estar en la carpeta *src*.
* Ayuda a los desarrolladores a determinar donde debe ir cada fichero.
* Permite crear scripts de construcción estandarizados.

##### Recomendación estructura general directorios del CD

**NOTA:** El nombre del CD debe corresponder con el del proyecto.

|  |  |
| --- | --- |
| **Directorio** | **Contenido** |
| *./ Directorio raíz del CD* | Contiene un fichero leeme.txt explicando toda esta estructura. Se puede incluir *autorun* e icono del proyecto si existe. |
| *./<nombre\_proyecto>* | Contiene toda la estructura de directorios del proyecto para desarrollo. **Ver la tabla Recomendación de estructura de directorios de desarrollo**. <nombre\_proyecto> debe sustituirse por el nombre corto del proyecto. |
| *./instalación* | Ficheros utilizados para la instalación del proyecto. |
| *./documentación* | Contiene toda la documentación asociada al proyecto. Es necesario incluir un fichero con el documento final del proyecto (en formato .doc o .docx de *Word* o bien .sxw de *Open Office*, por ejemplo) además de un fichero .PDF |
| *./documentacion/img* | Directorio que contiene las imágenes utilizadas en la documentación. Estas imágenes tendrán formato .png si son capturas de pantalla, .wmf si son diagramas o esquemas y .jpg sólo si son fotografías. |
| *./documentacion/uml* | Ficheros que genera la herramienta (Rose, *ArgoUML*, etc.) con la que se han generado los diagramas UML y de entidad relación. |
| *./presentación* | Directorio que contiene la presentación en *Powerpoint* o equivalente utilizada el día de la defensa del proyecto, si está disponible en el momento de realizar el CD. |
| *./herram* | Contiene los ficheros de instalación de las herramientas utilizadas para el desarrollo o puesta en marcha del proyecto (lógicamente sólo las que sean distribuibles). |
| *./herram/desarrollo* | Ficheros de instalación de las herramientas utilizadas en el desarrollo |
| *./herram/explotación* | BD, servidor Web y herramientas en general. |

##### Recomendación de la Estructura de Directorios de “desarrollo”

Se muestra aquí el contenido del directorio de desarrollo de la tabla anterior, incluyendo todos los directorios que deben depender del mismo. Algunos de los elementos sea han incluido suponiendo que se están usando ciertas tecnologías Java. En caso de no usarlas, buscaremos un equivalente existente (si lo hay) en la que estemos usando nosotros.

|  |  |
| --- | --- |
| **Directorio** | **Contenido** |
| *./ Directorio raíz de “desarrollo”* | Contiene los ficheros de proyecto del IDE utilizado. |
| *./build* | Contiene el build.xml de *ant* (si lo usamos). Debemos situarnos dentro para poder invocarlo. |
| *./conf* | Contiene los diferentes ficheros de configuración del proyecto. Podría contener distintos subdirectorios, en función de la tecnología usada. En este ejemplo se muestra un ejemplo de un proyecto *Web* hecho con tecnologías *Java*:   * **web**: contiene los ficheros de configuración de la aplicación Web (por ejemplo: web.xml). * **ear**: contiene los ficheros de configuración de una aplicación empresarial (por ejemplo: application.xml). * **ejb**: contiene los descriptores de despliegue de los EJB. |
| *./dist* | Directorio donde se sitúan los ficheros para la distribución del proyecto. Por ejemplo: los ficheros *.war* o *.ear.* |
| *./doc* | Contiene toda la documentación relativa al proyecto, incluyendo los ficheros generados por herramientas de generación de documentación automática como javadoc o similar. |
| *./lib* | Bibliotecas externas (.jar, .dll, …) necesarias para compilar y distribuir, de las que depende este proyecto. |
| *./compile-lib* | Bibliotecas externas (.jar, .dll, …) necesarias para compilar pero que no deseamos distribuir. |
| *./src* | Ficheros fuente |
| *./src/java* | Todos los ficheros *Java*, lógicamente agrupados en los paquetes correspondientes. |
| *./src/sql* | Este directorio contiene los scripts de *SQL* que permiten construir y meter los datos iniciales en la base de datos del proyecto (si existe). |
| *./web* | Este directorio contiene los ficheros (.JSP, .ASPX, .HTML, …) de la Web (si el proyecto incluyese una). |
| ./web/images | Contiene las imágenes utilizadas por los ficheros de la web del proyecto. |
| ./clases | Directorio donde se guardan los ficheros compilados (como por ejemplo los .class) |
| ./test | Directorio base para todos los ficheros utilizados en la automatización del proceso de prueba. |
| ./test/java | Contiene todas las pruebas unitarias utilizadas en el proceso de prueba automatizado. |
| ./test/sql | Scripts SQL utilizados en la carga de datos de prueba |
| ./bak | Directorio donde se pueden guardar versiones antiguas de los ficheros fuente del proyecto. |

### Código Ejecutable e Instalación

Descripción de los contenidos del código ejecutable y de la instalación de la aplicación en un ordenador. Breve manual de instalación y puesta en marcha de la aplicación (solamente unos pasos sencillos que faciliten este proceso, sin explicar nada (para eso está el manual de instalación propiamente dicho).

### Ficheros de Configuración

Descripción de todos los ficheros necesarios para poder hacer funcionar la aplicación (ficheros de configuración, ficheros de datos, etc.).

## Índice Alfabético

Este índice alfabético esta generado automáticamente por Word e incluirá todos los términos que nosotros marquemos adecuadamente con la herramienta que Word posee para ello. No debemos pues incluir palabras “a mano” en él. Este índice tiene una serie de ejemplos para ilustrar como funciona. No debemos olvidarnos de usar la opción “Actualizar campos” al finalizar de la documentación.

**NOTA: Quitar esta explicación en la documentación final.**

I

índice alfabético, 59

P

Palabra1, 5

problemas encontrados, 44

R

Redondo L., J. Manuel, 56

## Código Fuente

El código fuente tiene que ir dividido por paquetes y por archivos, con un formato que haga que resulte legible, tal y como se muestra a continuación en este ejemplo. En la mayoría de las ocasiones, es posible que no sea conveniente colocar la totalidad del código fuente en esta sección, sino solo una parte, que contenga aquellas clases más significativas e importantes. En cualquier caso, conviene consultar al director del proyecto este aspecto.

### Paquete Ejemplo 1:

#### Fichero “A.cs”:

using System;

using System.Collections.Generic;

using System.Text;

namespace ControlesAvisos

{

public class Usuario: ICloneable

{

public Usuario()

{

this.nombre = "";

this.clave = "";

}

public Usuario(string nombre, string clave)

{

this.nombre = nombre;

this.clave = clave;

}

private string nombre;

public string Nombre

{

get

{

return nombre;

}

set

{

nombre = value;

}

}

private string clave;

public string Clave

{

get

{

return clave;

}

set

{

clave = value;

}

}

public Object Clone()

{

Usuario u = new Usuario(this.Nombre, this.Clave);

return u;

}

}

}

1. No necesariamente debe ser una única palabra, pueden ser varias. Por ejemplo, si el proyecto tratase sobre la gestión de calificaciones de Alumnos, una palabra clave válida podría ser “Expediente Académico”. [↑](#footnote-ref-1)